Semantic Parsing with Semi-Supervised Sequential Autoencoders

Tomáš Kočiský[†‡] **Gábor Melis**[†] **Edward Grefenstette**[†]
Chris Dyer[†] **Wang Ling**[†] **Phil Blunsom**[†‡] **Karl Moritz Hermann**[†]
[†]Google DeepMind [‡]University of Oxford
{tkocisky,melisgl,etg,cdyer,lingwang,pblunsom,kmh}@google.com

Abstract

We present a novel semi-supervised approach for sequence transduction and apply it to semantic parsing. The unsupervised component is based on a generative model in which latent sentences generate the unpaired logical forms. We apply this method to a number of semantic parsing tasks focusing on domains with limited access to labelled training data and extend those datasets with synthetically generated logical forms.

1 Introduction

Neural approaches, in particular attention-based sequence-to-sequence models, have shown great promise and obtained state-of-the-art performance for sequence transduction tasks including machine translation (Bahdanau et al., 2015), syntactic constituency parsing (Vinyals et al., 2015), and semantic role labelling (Zhou and Xu, 2015). A key requirement for effectively training such models is an abundance of supervised data.

In this paper we focus on learning mappings from input sequences x to output sequences y in domains where the latter are easily obtained, but annotation in the form of (x, y) pairs is sparse or expensive to produce, and propose a novel architecture that accommodates semi-supervised training on sequence transduction tasks. To this end, we augment the transduction objective ($x \mapsto y$) with an autoencoding objective where the input sequence is treated as a latent variable ($y \mapsto x \mapsto y$), enabling training from both labelled pairs and unpaired output sequences. This is common in situations where we encode natural language into a logical form governed by some grammar or database.

While such an autoencoder could in principle be constructed by stacking two sequence transducers, modelling the latent variable as a series of discrete symbols drawn from multinomial distributions creates serious computational challenges, as it requires marginalising over the space of latent sequences Σ_x^*. To avoid this intractable marginalisation, we introduce a novel differentiable alternative for draws from a softmax which can be used with the reparametrisation trick of Kingma and Welling (2014). Rather than drawing a discrete symbol in Σ_x from a softmax, we draw a distribution over symbols from a logistic-normal distribution at each time step. These serve as continuous relaxations of discrete samples, providing a differentiable estimator of the expected reconstruction log likelihood.

We demonstrate the effectiveness of our proposed model on three semantic parsing tasks: the GEOQUERY benchmark (Zelle and Mooney, 1996; Wong and Mooney, 2006), the SAIL maze navigation task (MacMahon et al., 2006) and the Natural Language Querying corpus (Haas and Riezler, 2016) on OpenStreetMap. As part of our evaluation, we introduce simple mechanisms for generating large amounts of unsupervised training data for two of these tasks.

In most settings, the semi-supervised model outperforms the supervised model, both when trained on additional generated data as well as on subsets of the existing data.

Dataset	Example
GEO	what are the high points of states surrounding mississippi answer(high_point_1(state(next_to_2(stateid('mississippi')))))
NLMAPS	Where are kindergartens in Hamburg? query(area(keyval('name','Hamburg')),nwr(keyval('amenity','kindergarten')),qtype(latlong))
SAIL	turn right at the bench into the yellow tiled hall $(1, 6, 90)$ FORWARD - FORWARD - RIGHT - STOP $(3, 6, 180)$

Table 1: Examples of natural language x and logical form y from the three corpora and tasks used in this paper. Note that the SAIL corpus requires additional information in order to map from the instruction to the action sequence.

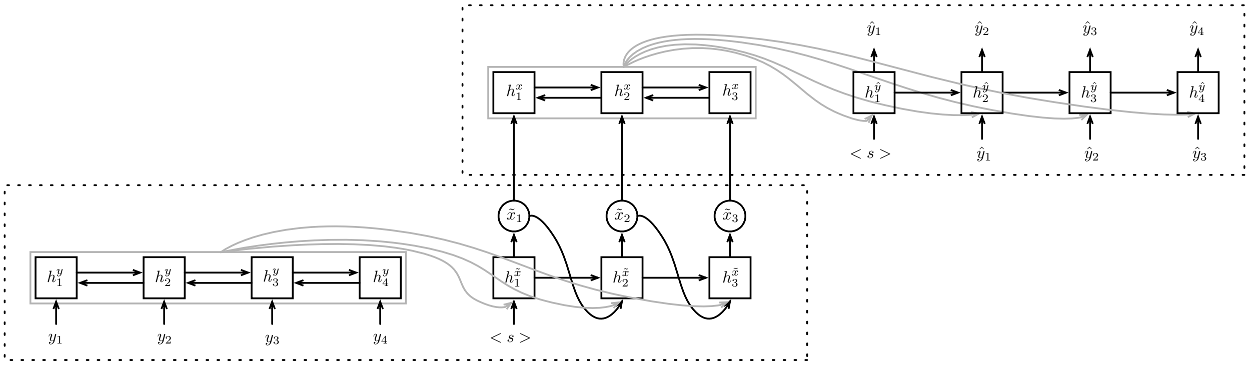

Figure 1: SEQ4 model with attention-sequence-to-sequence encoder and decoder. Circle nodes represent random variables.

2 Model

Our sequential autoencoder is shown in Figure 1. At a high level, it can be seen as two sequence-to-sequence models with attention (Bahdanau et al., 2015) chained together. More precisely, the model consists of four LSTMs (Hochreiter and Schmidhuber, 1997), hence the name SEQ4. The first, a bidirectional LSTM, encodes the sequence y; next, an LSTM with stochastic output, described below, draws a sequence of distributions $\tilde{x}$ over words in vocabulary Σ_x. The third LSTM encodes these distributions for the last one to attend over and reconstruct y as $\hat{y}$. We now give the details of these parts.

2.1 Encoding y

The first LSTM of the encoder half of the model reads the sequence y, represented as a sequence of one-hot vectors over the vocabulary Σ_y, using a bidirectional RNN into a sequence of vectors $h^y_{1:L_y}$ where L_y is the sequence length of y,

$$h^y_t = \left(f^{\rightarrow}_y(y_t, h^{y,\rightarrow}_{t-1}); f^{\leftarrow}_y(y_t, h^{y,\leftarrow}_{t+1})\right), \qquad (1)$$

where $f^{\rightarrow}_y, f^{\leftarrow}_y$ are non-linear functions applied at each time step to the current token y_t and their recurrent states $h^{y,\rightarrow}_{t-1}$, $h^{y,\leftarrow}_{t+1}$, respectively.

Both the forward and backward functions project the one-hot vector into a dense vector via an embedding matrix, which serves as input to an LSTM.

2.2 Predicting a Latent Sequence $\tilde{x}$

Subsequently, we wish to predict x. Predicting a discrete sequence of symbols through draws from multinomial distributions over a vocabulary is not an option, as we would not be able to backpropagate through this discrete choice. Marginalising over the possible latent strings or estimating the gradient through naïve Monte Carlo methods would be a prohibitively high variance process because the number of strings is exponential in the maximum length (which we would have to manually specify) with the vocabulary size as base. To allow backpropagation, we instead predict a sequence of distributions $\tilde{x}$ over the symbols of Σ_x with an RNN attending over

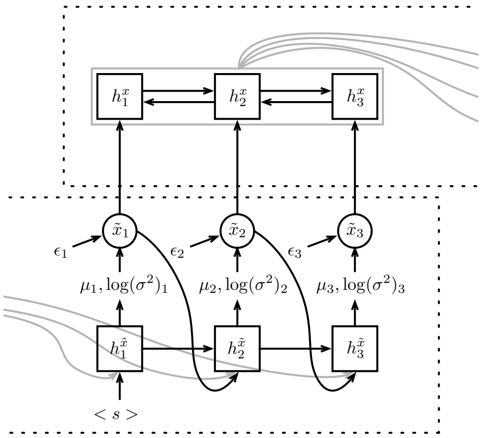

Figure 2: Unsupervised case of the SEQ4 model.

$h^y = h^y_{1:L_y}$, which will later serve to reconstruct y:

$$\tilde{x} = q(x|y) = \prod_{t=1}^{L_x} q(\tilde{x}_t | \{\tilde{x}_1, \cdots, \tilde{x}_{t-1}\}, h^y) \quad (2)$$

where $q(x|y)$ models the mapping $y \mapsto x$. We define $q(\tilde{x}_t | \{\tilde{x}_1, \cdots, \tilde{x}_{t-1}\}, h^y)$ in the following way:

Let the vector $\tilde{x}_t$ be a distribution over the vocabulary Σ_x drawn from a logistic-normal distribution[1], the parameters of which, $\mu_t, \log(\sigma^2)_t \in \mathbb{R}^{|\Sigma_x|}$, are predicted by attending by an LSTM attending over the outputs of the encoder (Equation 2), where $|\Sigma_x|$ is the size of the vocabulary Σ_x. The use of a logistic normal distribution serves to regularise the model in the semi-supervised learning regime, which is described at the end of this section. Formally, this process, depicted in Figure 2, is as follows:

$$h_t^{\tilde{x}} = f_{\tilde{x}}(\tilde{x}_{t-1}, h_{t-1}^{\tilde{x}}, h^y) \quad (3)$$

$$\mu_t, \log(\sigma_t^2) = l(h_t^{\tilde{x}}) \quad (4)$$

$$\epsilon \sim \mathcal{N}(0, I) \quad (5)$$

$$\gamma_t = \mu_t + \sigma_t \epsilon \quad (6)$$

$$\tilde{x}_t = \text{softmax}(\gamma_t) \quad (7)$$

where the $f_{\tilde{x}}$ function is an LSTM and l a linear transformation to $\mathbb{R}^{2|\Sigma_x|}$. We use the reparametrisation trick from Kingma and Welling (2014) to draw from the logistic normal, allowing us to backpropagate through the sampling process.

2.3 Encoding x

Moving on to the decoder part of our model, in the third LSTM, we embed[2] and encode $\tilde{x}$:

$$h_t^x = \left(f_x^{\rightarrow}(\tilde{x}_t, h_{t-1}^{x,\rightarrow}); f_x^{\leftarrow}(\tilde{x}_t, h_{t+1}^{x,\leftarrow})\right) \quad (8)$$

When x is observed, during supervised training and also when making predictions, instead of the distribution $\tilde{x}$ we feed the one-hot encoded x to this part of the model.

2.4 Reconstructing y

In the final LSTM, we decode into y:

$$p(\hat{y}|\tilde{x}) = \prod_{t=1}^{L_y} p(\hat{y}_t | \{\hat{y}_1, \cdots, \hat{y}_{t-1}\}, h^{\tilde{x}}) \quad (9)$$

Equation 9 is implemented as an LSTM attending over $h^{\tilde{x}}$ producing a sequence of symbols $\hat{y}$ based on recurrent states $h^{\hat{y}}$, aiming to reproduce input y:

$$h_t^{\hat{y}} = f_{\hat{y}}(\hat{y}_{t-1}, h_{t-1}^{\hat{y}}, h^{\tilde{x}}) \quad (10)$$

$$\hat{y}_t \sim \text{softmax}(l'(h_t^{\hat{y}})) \quad (11)$$

where $f_{\hat{y}}$ is the non-linear function, and the actual probabilities are given by a softmax function after a linear transformation l' of $h^{\hat{y}}$. At training time, rather than $\hat{y}_{t-1}$ we feed the ground truth y_{t-1}.

2.5 Loss function

The complete model described in this section gives a reconstruction function $y \mapsto \hat{y}$. We define a loss on this reconstruction which accommodates the unsupervised case, where x is not observed in the training data, and the supervised case, where (x, y) pairs are available. Together, these allow us to train the SEQ4 model in a semi-supervised setting, which experiments will show provides some benefits over a purely supervised training regime.

Unsupervised case When x isn't observed, the loss we minimise during training is the reconstruction loss on y, expressed as the negative log-likelihood $NLL(\hat{y}, y)$ of the true labels y relative to the predictions $\hat{y}$. To this, we add as a regularising

[1] The logistic-normal distribution is the exponentiated and normalised (i.e. taking softmax) normal distribution.

[2] Multiplying the distribution over words and an embedding matrix averages the word embedding of the entire vocabulary weighted by their probabilities.

term the KL divergence $KL[q(\gamma|y)\|p(\gamma)]$ which effectively penalises the mean and variance of $q(\gamma|y)$ from diverging from those of a prior $p(\gamma)$, which we model as a diagonal Gaussian $\mathcal{N}(0, I)$. This has the effect of smoothing the logistic normal distribution from which we draw the distributions over symbols of x, guarding against overfitting of the latent distributions over x to symbols seen in the supervised case discussed below. The unsupervised loss is therefore formalised as

$$\mathcal{L}_{unsup} = NLL(\hat{y}, y) + \alpha KL[q(\gamma|y)\|p(\gamma)] \quad (12)$$

with regularising factor α is tuned on validation, and

$$KL[q(\gamma|y)\|p(\gamma)] = \sum_{i=1}^{L_x} KL[q(\gamma_i|y)\|p(\gamma)] \quad (13)$$

We use a closed form of these individual KL divergences, described by Kingma and Welling (2014).

Supervised case When x is observed, we additionally minimise the prediction loss on x, expressed as the negative log-likelihood $NLL(\tilde{x}, x)$ of the true labels x relative to the predictions $\tilde{x}$, and do not impose the KL loss. The supervised loss is thus

$$\mathcal{L}_{sup} = NLL(\tilde{x}, x) + NLL(\hat{y}, y) \quad (14)$$

In both the supervised and unsupervised case, because of the continuous relaxation on generating $\tilde{x}$ and the reparameterisation trick, the gradient of the losses with regard to the model parameters is well defined throughout SEQ4.

Semi-supervised training and inference We train with a weighted combination of the supervised and unsupervised losses described above. Once trained, we simply use the $x \mapsto y$ decoder segment of the model to predict y from sequences of symbols x represented as one-hot vectors. When the decoder is trained without the encoder in a fully supervised manner, it serves as our supervised sequence-to-sequence baseline model under the name S2S.

3 Tasks and Data Generation

We apply our model to three tasks outlined in this section. Moreover, we explain how we generated additional unsupervised training data for two of these tasks. Examples from all datasets are in Table 1.

3.1 GeoQuery

The first task we consider is the prediction of a query on the GEO corpus which is a frequently used benchmark for semantic parsing. The corpus contains 880 questions about US geography together with executable queries representing those questions. We follow the approach established by Zettlemoyer and Collins (2005) and split the corpus into 600 training and 280 test cases. Following common practice, we augment the dataset by referring to the database during training and test time. In particular, we use the database to identify and anonymise variables (cities, states, countries and rivers) following the method described in Dong and Lapata (2016).

Most prior work on the GEO corpus relies on standard semantic parsing methods together with custom heuristics or pipelines for this corpus. The recent paper by Dong and Lapata (2016) is of note, as it uses a sequence-to-sequence model for training which is the unidirectional equivalent to S2S, and also to the decoder part of our SEQ4 network.

3.2 Open Street Maps

The second task we tackle with our model is the NLMAPS dataset by Haas and Riezler (2016). The dataset contains 1,500 training and 880 testing instances of natural language questions with corresponding machine readable queries over the geographical OpenStreetMap database. The dataset contains natural language question in both English and German but we focus only on single language semantic parsing, similar to the first task in Haas and Riezler (2016). We use the data as it is, with the only pre-processing step being the tokenization of both natural language and query form[3].

3.3 Navigational Instructions to Actions

The SAIL corpus and task were developed to train agents to follow free-form navigational route instructions in a maze environment (MacMahon et al., 2006; Chen and Mooney, 2011). It consists of a small number of mazes containing features such as objects, wall and floor types. These mazes come together with a large number of human instructions paired with the required actions[4] to reach the goal

[3] We removed quotes, added spaces around `()`, and separated the question mark from the last word in each question.

[4] There are four actions: `LEFT`, `RIGHT`, `GO`, `STOP`.

state described in those instructions.

We use the sentence-aligned version of the SAIL route instruction dataset containing 3,236 sentences (Chen and Mooney, 2011). Following previous work, we accept an action sequence as correct if and only if the final position and orientation exactly match those of the gold data. We do not perform any pre-processing on this dataset.

3.4 Data Generation

As argued earlier, we are focusing on tasks where aligned data is sparse and expensive to obtain, while it should be cheap to get unsupervised, monomodal data. Albeit that is a reasonable assumption for real world data, the datasets considered have no such component, thus the approach taken here is to generate random database queries or maze paths, i.e. the machine readable side of the data, and train a semi-supervised model. The alternative not explored here would be to generate natural language questions or instructions instead, but that is more difficult to achieve without human intervention. For this reason, we generate the machine readable side of the data for GEOQUERY and SAIL tasks[5].

For GEOQUERY, we fit a 3-gram Kneser-Ney (Chen and Goodman, 1999) model to the queries in the training set and sample about 7 million queries from it. We ensure that the sampled queries are different from the training queries, but do not enforce validity. This intentionally simplistic approach is to demonstrate the applicability of our model.

The SAIL dataset has only three mazes. We added a fourth one and over 150k random paths, including duplicates. The new maze is larger (21 × 21 grid) than the existing ones, and seeks to approximately replicate the key statistics of the other three mazes (maximum corridor length, distribution of objects, etc). Paths within that maze are created by randomly sampling start and end positions.

4 Experiments

We evaluate our model on the three tasks in multiple settings. First, we establish a supervised baseline to compare the S2S model with prior work. Next, we train our SEQ4 model in a semi-supervised setting on the entire dataset with the additional monomodal training data described in the previous section.

Model	Accuracy
Zettlemoyer and Collins (2005)	79.3
Zettlemoyer and Collins (2007)	86.1
Liang et al. (2013)	87.9
Kwiatkowski et al. (2011)	88.6
Zhao and Huang (2014)	88.9
Kwiatkowski et al. (2013)	89.0
Dong and Lapata (2016)	84.6
Jia and Liang (2016)[6]	89.3
S2S	86.5
SEQ4	87.3

Table 2: Non-neural and neural model results on GEOQUERY using the train/test split from (Zettlemoyer and Collins, 2005).

Finally, we perform an "ablation" study where we discard some of the training data and compare S2S to SEQ4. S2S is trained solely on the reduced data in a supervised manner, while SEQ4 is once again trained semi-supervised on the same reduced data plus the machine readable part of the discarded data (SEQ4-) or on the extra generated data (SEQ4+).

Training We train the model using standard gradient descent methods. As none of the datasets used here contain development sets, we tune hyperparameters by cross-validating on the training data. In the case of the SAIL corpus we train on three folds (two mazes for training and validation, one for test each) and report weighted results across the folds following prior work (Mei et al., 2016).

4.1 GeoQuery

The evaluation metric for GEOQUERY is the accuracy of exactly predicting the machine readable query. As results in Table 2 show, our supervised S2S baseline model performs slightly better than the comparable model by Dong and Lapata (2016). The semi-supervised SEQ4 model with the additional generated queries improves on it further.

The ablation study in Table 3 demonstrates a widening gap between supervised and semi-

[5] Our randomly generated unsupervised datasets can be downloaded from `http://deepmind.com/publications`

[6] Jia and Liang (2016) used hand crafted grammars to generate additional supervised training data.

Sup. data	S2S	Seq4-	Seq4+
5%	21.9	30.1	26.2
10%	39.7	42.1	42.1
25%	62.4	70.4	67.1
50%	80.3	81.2	80.4
75%	85.3	84.1	85.1
100%	86.5	86.5	87.3

Table 3: Results of the GeoQuery ablation study.

Model	Accuracy
Haas and Riezler (2016)	68.30
S2S	78.03

Table 4: Results on the NLmaps corpus.

supervised as the amount of labelled training data gets smaller. This suggests that our model can leverage unlabelled data even when only small amount of labelled data is available.

4.2 Open Street Maps

We report results for the NLmaps corpus in Table 4, comparing the supervised S2S model to the results posted by Haas and Riezler (2016). While their model used a semantic parsing pipeline including alignment, stemming, language modelling and CFG inference, the strong performance of the S2S model demonstrates the strength of fairly vanilla attention-based sequence-to-sequence models. It should be pointed out that the previous work reports the number of correct answers when queries were executed against the dataset, while we evaluate on the strict accuracy of the generated queries. While we expect these numbers to be nearly equivalent, our evaluation is strictly harder as it does not allow for reordering of query arguments and similar relaxations.

We investigate the Seq4 model only via the ablation study in Table 5 and find little gain through the semi-supervised objective. Our attempt at cheaply generating unsupervised data for this task was not successful, likely due to the complexity of the underlying database.

4.3 Navigational Instructions to Actions

Model extension The experiments for the SAIL task differ slightly from the other two tasks in that the language input does not suffice for choosing an action. While a simple instruction such as *'turn left'* can easily be translated into the action sequence `LEFT-STOP`, more complex instructions such as *'Walk forward until you see a lamp'* require knowledge of the agent's position in the maze.

Sup. data	S2S	Seq4-
5%	3.22	3.74
10%	17.61	17.12
25%	33.74	33.50
50%	49.52	53.72
75%	66.93	66.45
100%	78.03	78.03

Table 5: Results of the NLmaps ablation study.

To accomplish this we modify the model as follows. First, when encoding action sequences, we concatenate each action with a representation of the maze at the given position, representing the maze-state akin to Mei et al. (2016) with a bag-of-features vector. Second, when decoding action sequences, the RNN outputs an action which is used to update the agent's position and the representation of that new position is fed into the RNN as its next input.

Training regime We cross-validate over the three mazes in the dataset and report overall results weighted by test size (cf. Mei et al. (2016)). Both our supervised and semi-supervised model perform worse than the state-of-the-art (see Table 6), but the latter enjoys a comfortable margin over the former. As the S2S model broadly reimplements the work of Mei et al. (2016), we put the discrepancy in performance down to the particular design choices that we did not follow in order to keep the model here as general as possible and comparable across tasks.

The ablation studies (Table 7) show little gain for the semi-supervised approach when only using data from the original training set, but substantial improvement with the additional unsupervised data.

5 Discussion

Supervised training The prediction accuracies of our supervised baseline S2S model are mixed with respect to prior results on their respective tasks. For GeoQuery, S2S performs significantly better than the most similar model from the literature (Dong and Lapata, 2016), mostly due to the fact that y and x are

Input from unsupervised data (y)	**Generated latent representation** (x)
answer smallest city loc_2 state stateid _STATE_	what is the smallest city in the state of _STATE_ </S>
answer city loc_2 state next_to_2 stateid _STATE_	what are the cities in states which border _STATE_ </S>
answer mountain loc_2 countryid _COUNTRY_	what is the lakes in _COUNTRY_ </S>
answer state next_to_2 state all	which states longer states show peak states to </S>

Table 8: Positive and negative examples of latent language together with the randomly generated logical form from the unsupervised part of the GeoQuery training. Note that the natural language (x) does not occur anywhere in the training data in this form.

Model	Accuracy
Chen and Mooney (2011)	54.40
Kim and Mooney (2012)	57.22
Andreas and Klein (2015)	59.60
Kim and Mooney (2013)	62.81
Artzi et al. (2014)	64.36
Artzi and Zettlemoyer (2013)	65.28
Mei et al. (2016)	69.98
S2S	58.60
Seq4	63.25

Table 6: Results on the SAIL corpus.

Sup. data	S2S	Seq4-	Seq4+
5%	37.79	41.48	43.44
10%	40.77	41.26	48.67
25%	43.76	43.95	51.19
50%	48.01	49.42	55.97
75%	48.99	49.20	57.40
100%	49.49	49.49	58.28

Table 7: Results of the SAIL ablation study. Results are from models trained on *L* and *Jelly* maps, tested on *Grid* only, hence the discrepancy between the 100% result and S2S in Table 6.

encoded with bidirectional LSTMs. With a unidirectional LSTM we get similar results to theirs.

On the SAIL corpus, S2S performs worse than the state of the art. As the models are broadly equivalent we attribute this difference to a number of task-specific choices and optimisations[7] made in Mei et al. (2016) which we did not reimplement for the sake of using a common model across all three tasks.

For NLmaps, S2S performs much better than the state-of-the-art, exceeding the previous best result by 11% despite a very simple tokenization method and a lack of any form of entity anonymisation.

Semi-supervised training In both the case of GeoQuery and the SAIL task we found the semi-supervised model to convincingly outperform the fully supervised model. The effect was particularly notable in the case of the SAIL corpus, where performance increased from 58.60% accuracy to 63.25% (see Table 6). It is worth remembering that the supervised training regime consists of three folds of tuning on two maps with subsequent testing on the third map, which carries a risk of overfitting to the training maps. The introduction of the fourth unsupervised map clearly mitigates this effect. Table 8 shows some examples of unsupervised logical forms being transformed into natural language, which demonstrate how the model can learn to sensibly ground unsupervised data.

Ablation performance The experiments with additional unsupervised data prove the feasibility of our approach and clearly demonstrate the usefulness of the Seq4 model for the general class of sequence-to-sequence tasks where supervised data is hard to come by. To analyse the model further, we also look at the performance of both S2S and Seq4 when reducing the amount of supervised training data available to the model. We compare three settings: the supervised S2S model with reduced training data, Seq4- which uses the removed training data in an unsupervised fashion (throwing away the natural language) and Seq4+ which uses the randomly generated unsupervised data described in Section 3. The S2S model behaves as expected on all three tasks, its performance dropping with the size of the training data. The performance of Seq4- and Seq4+ requires more analysis.

In the case of GeoQuery, having unlabelled data from the true distribution (Seq4-) is a good thing

[7] In particular we don't use beam search and ensembling.

when there is enough of it, as clearly seen when only 5% of the original dataset is used for supervised training and the remaining 95% is used for unsupervised training. The gap shrinks as the amount of supervised data is increased, which is as expected. On the other hand, using a large amount of extra, generated data from an approximating distribution (SEQ4+) does not help as much initially when compared with the unsupervised data from the true distribution. However, as the size of the unsupervised dataset in SEQ4- becomes the bottleneck this gap closes and eventually the model trained on the extra data achieves higher accuracy.

For the SAIL task the semi-supervised models do better than the supervised results throughout, with the model trained on randomly generated additional data consistently outperforming the model trained only on the original data. This gives further credence to the risk of overfitting to the training mazes already mentioned above.

Finally, in the case of the NLMAPS corpus, the semi-supervised approach does not appear to help much at any point during the ablation. These indistinguishable results are likely due to the task's complexity, causing the ablation experiments to either have to little supervised data to sufficiently ground the latent space to make use of the unsupervised data, or in the higher percentages then too little unsupervised data to meaningfully improve the model.

6 Related Work

Semantic parsing The tasks in this paper all broadly belong to the domain of semantic parsing, which describes the process of mapping natural language to a formal representation of its meaning. This is extended in the SAIL navigation task, where the formal representation is a function of both the language instruction and a given environment.

Semantic parsing is a well-studied problem with numerous approaches including inductive logic programming (Zelle and Mooney, 1996), string-to-tree (Galley et al., 2004) and string-to-graph (Jones et al., 2012) transducers, grammar induction (Kwiatkowski et al., 2011; Artzi and Zettlemoyer, 2013; Reddy et al., 2014) or machine translation (Wong and Mooney, 2006; Andreas et al., 2013).

While a large number of relevant literature focuses on defining the grammar of the logical forms (Zettlemoyer and Collins, 2005), other models learn purely from aligned pairs of text and logical form (Berant and Liang, 2014), or from more weakly supervised signals such as question-answer pairs together with a database (Liang et al., 2011). Recent work of Jia and Liang (2016) induces a synchronous context-free grammar and generates additional training examples (x, y), which is one way to address data scarcity issues. The semi-supervised setup proposed here offers an alternative solution to this issue.

Discrete autoencoders Very recently there has been some related work on discrete autoencoders for natural language processing (Suster et al., 2016; Marcheggiani and Titov, 2016, *i.a.*) This work presents a first approach to using effectively discretised sequential information as the latent representation without resorting to draconian assumptions (Ammar et al., 2014) to make marginalisation tractable. While our model is not exactly marginalisable either, the continuous relaxation makes training far more tractable. A related idea was recently presented in Gülçehre et al. (2015), who use monolingual data to improve machine translation by fusing a sequence-to-sequence model and a language model.

7 Conclusion

We described a method for augmenting a supervised sequence transduction objective with an autoencoding objective, thereby enabling semi-supervised training where previously a scarcity of aligned data might have held back model performance. Across multiple semantic parsing tasks we demonstrated the effectiveness of this approach, improving model performance by training on randomly generated unsupervised data in addition to the original data.

Going forward it would be interesting to further analyse the effects of sampling from a logistic-normal distribution as opposed to a softmax in order to better understand how this impacts the distribution in the latent space. While we focused on tasks with little supervised data and additional unsupervised data in y, it would be straightforward to reverse the model to train it with additional labelled data in x, i.e. on the natural language side. A natural extension would also be a formulation where semi-supervised training was performed in both x and y.

For instance, machine translation lends itself to such a formulation where for many language pairs parallel data may be scarce while there is an abundance of monolingual data.

References

Waleed Ammar, Chris Dyer, and Noah A. Smith. 2014. Conditional Random Field Autoencoders for Unsupervised Structured Prediction. In *Proceedings of NIPS*.

Jacob Andreas and Dan Klein. 2015. Alignment-based Compositional Semantics for Instruction Following. In *Proceedings of EMNLP*, September.

Jacob Andreas, Andreas Vlachos, and Stephen Clark. 2013. Semantic Parsing as Machine Translation. In *Proceedings of ACL*, August.

Yoav Artzi and Luke Zettlemoyer. 2013. Weakly Supervised Learning of Semantic Parsers for Mapping Instructions to Actions. *Transactions of the Association for Computational Linguistics*, 1(1):49–62.

Yoav Artzi, Dipanjan Das, and Slav Petrov. 2014. Learning Compact Lexicons for CCG Semantic Parsing. In *Proceedings of EMNLP*, October.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *Proceedings of ICLR*.

Jonathan Berant and Percy Liang. 2014. Semantic Parsing via Paraphrasing. In *Proceedings of ACL*, June.

Stanley F Chen and Joshua Goodman. 1999. An empirical study of smoothing techniques for language modeling. *Computer Speech & Language*, 13(4):359–393.

David L. Chen and Raymond J. Mooney. 2011. Learning to Interpret Natural Language Navigation Instructions from Observations. In *Proceedings of AAAI*, August.

Li Dong and Mirella Lapata. 2016. Language to Logical Form with Neural Attention. *arXiv preprint arXiv:1601.01280*.

Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. 2004. What's in a translation rule? In *Proceedings of HLT-NAACL*, May.

Çaglar Gülçehre, Orhan Firat, Kelvin Xu, Kyunghyun Cho, Loïc Barrault, Huei-Chi Lin, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2015. On Using Monolingual Corpora in Neural Machine Translation. *arXiv preprint arXiv:1503.03535*.

Carolin Haas and Stefan Riezler. 2016. A corpus and semantic parser for multilingual natural language querying of openstreetmap. In *Proceedings of NAACL*, June.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, November.

Robin Jia and Percy Liang. 2016. Data recombination for neural semantic parsing. In *Association for Computational Linguistics (ACL)*.

Bevan Jones, Jacob Andreas, Daniel Bauer, Karl Moritz Hermann, and Kevin Knight. 2012. Semantics-Based Machine Translation with Hyperedge Replacement Grammars. In *Proceedings of COLING 2012*, December.

Joohyun Kim and Raymond J. Mooney. 2012. Unsupervised PCFG Induction for Grounded Language Learning with Highly Ambiguous Supervision. In *Proceedings of EMNLP-CoNLL*, July.

Joohyun Kim and Raymond Mooney. 2013. Adapting Discriminative Reranking to Grounded Language Learning. In *Proceedings of ACL*, August.

Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *Proceedings of ICLR*.

Tom Kwiatkowski, Luke Zettlemoyer, Sharon Goldwater, and Mark Steedman. 2011. Lexical Generalization in CCG Grammar Induction for Semantic Parsing. In *Proceedings of EMNLP*.

Tom Kwiatkowski, Eunsol Choi, Yoav Artzi, and Luke Zettlemoyer. 2013. Scaling semantic parsers with on-the-fly ontology matching. In *In Proceedings of EMNLP*. Citeseer.

Percy Liang, Michael I. Jordan, and Dan Klein. 2011. Learning Dependency-based Compositional Semantics. In *Proceedings of the ACL-HLT*.

Percy Liang, Michael I Jordan, and Dan Klein. 2013. Learning dependency-based compositional semantics. *Computational Linguistics*, 39(2):389–446.

Matt MacMahon, Brian Stankiewicz, and Benjamin Kuipers. 2006. Walk the Talk: Connecting Language, Knowledge, and Action in Route Instructions. In *Proceedings of AAAI*.

Diego Marcheggiani and Ivan Titov. 2016. Discrete-state variational autoencoders for joint discovery and factorization of relations. *Transactions of ACL*.

Hongyuan Mei, Mohit Bansal, and Matthew R. Walter. 2016. Listen, Attend, and Walk: Neural Mapping of Navigational Instructions to Action Sequences. In *Proceedings of AAAI*.

Siva Reddy, Mirella Lapata, and Mark Steedman. 2014. Large-scale Semantic Parsing without Question-Answer Pairs. *Transactions of the Association for Computational Linguistics*, 2:377–392.

Simon Suster, Ivan Titov, and Gertjan van Noord. 2016. Bilingual Learning of Multi-sense Embeddings with Discrete Autoencoders. *CoRR*, abs/1603.09128.

Oriol Vinyals, Łukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. 2015. Grammar as a Foreign Language. In *Proceedings of NIPS*.

Yuk Wah Wong and Raymond J. Mooney. 2006. Learning for Semantic Parsing with Statistical Machine Translation. In *Proceedings of NAACL*.

John M. Zelle and Raymond J. Mooney. 1996. Learning to Parse Database Queries using Inductive Logic Programming. In *Proceedings of AAAI/IAAI*, pages 1050–1055, August.

Luke S. Zettlemoyer and Michael Collins. 2005. Learning to Map Sentences to Logical Form: Structured Classification with Probabilistic Categorial Grammars. In *UAI*, pages 658–666. AUAI Press.

Luke Zettlemoyer and Michael Collins. 2007. Online Learning of Relaxed CCG Grammars for Parsing to Logical Form. In *Proceedings of EMNLP-CoNLL*, June.

Kai Zhao and Liang Huang. 2014. Type-driven incremental semantic parsing with polymorphism. *arXiv preprint arXiv:1411.5379*.

Jie Zhou and Wei Xu. 2015. End-to-end Learning of Semantic Role Labeling Using Recurrent Neural Networks. In *Proceedings of ACL*.